Symplectic multi-time step parareal algorithms applied to molecular dynamics

Christophe Audouze, Marc Massot, Sebastian Volz

SYMPLECTIC MULTI-TIME STEP PARAREAL ALGORITHMS APPLIED TO MOLECULAR DYNAMICS

CHRISTOPHE AUDOUZE, MARC MASSOT, SEBASTIAN VOLZ

Laboratoire EM2C, Ecole Centrale Paris
Grande Voie des Vignes, 92295 Châtenay-Malabry, France

Abstract. In this paper we propose a new parareal algorithm for parallelizing in time molecular dynamics problems. The original structure of this algorithm allows one to consider multi-time stepping, namely two levels of temporal discretization, providing a larger range for the fine and coarse solvers definition. We also prove the symplecticity of this method, which is an expected behavior of the molecular dynamics integrators. The relevance of this algorithm is numerically demonstrated by applying it to three-dimensional atomic lattices on parallel computer architectures. For lattices of more than 20000 atoms we get attractive speed-up with proper choice for the coarse solver definition and for the number of processors.

Key words. temporal parallelization, parareal algorithm, ODE integration, multi-time stepping, symplectic integration, molecular dynamics

AMS subject classifications. 65Y05, 65L06, 65P10, 65Y20, 65Z05

1. Introduction and motivation. For various applications in the field of material science, the determination of some physical properties such as thermal conductivity or the polarisability of electric or semi-conducting materials requires very costly numerical simulations. A way to get such properties consists in performing classical molecular dynamics (MD) simulations on a great number of atoms, typically several tens of thousands (see [1] for a general presentation of MD and [2, 3, 4] for MD applied to nanoscale heat transfer problems). The classical MD equations of motion have to be integrated during a long time to reach the thermodynamical equilibrium and deduce statistical quantities of interest such as temperature or pressure. More precisely the atomic oscillation period is about one picosecond, the time integration can reach hundreds of picoseconds with a time step of about one femtosecond. Moreover the nature of the potential used to compute the interatomic forces can also significantly increase the computation time: for example the Stillinger-Weber potential [1] typically used in low dimensional heat transfer applications contains a two-body interaction term as well as a very costly three-body angular potential contribution. Thanks to the rapidly growing development of high performance computing, techniques such as decomposition domain [5, 6, 7, 8, 9, 10] methods (DD) are successfully used on computers with massively parallel architectures, allowing to consider larger and larger atomic systems. As striking examples, biophysical simulations recently involve 10^6 atoms of a complete virus on an integration time of 50ns [5], while MD simulations of shockwave phenomena have been realized on several billions of atoms using computers with massively parallel architectures [11]. In this high performance computing context, it seems relevant to propose new algorithmic techniques in order to decrease the computational cost of simulations. Even if the spatial parallelization such as the DD technique has been used successfully in the MD framework and more generally in a large number of scientific areas such as fluids mechanics, the temporal parallelization is less commonly spread. Parallel algorithms for the time direction were first introduced in [12, 13, 14] for integrating Ordinary Differential Equations (ODE) while the parareal algorithm developed for Partial Differential Equations (PDE) problems has been proposed more recently in [15]. Nevertheless, the temporal direction would also need to be taken into

account carefully. Indeed the related ideas of the *parareal algorithms* developed for the temporal parallelization are rather recent and are not as intuitive as in the spatial parallelism.

The aim of the present paper is to propose a new parareal algorithm designed for MD problems based on classical mechanics and demonstrate the relevance of parallelizing the time direction. The use of these algorithms is motivated by the significant and helpful numerical analysis literature which exists on the subject [16, 17, 18]. Let us notice that the temporal parallelization has already been used for *ab-initio* MD problems [19]. As mentioned in this article, the symplectic feature of the parareal methods is not clear and needs to be investigated. Our paper answers to this algorithmic question providing a parareal algorithm which is proven to be symplectic. Another important point concerns the ability to use two levels of temporal discretization, corresponding to the *coarse* and *fine* resolutions in the parareal algorithm. The classical MD integrators based on the Verlet scheme [1] do not allow to use different levels of temporal discretization; our new parareal algorithm overcomes this difficulty. The *multi-stepping* is an important feature since it gives a larger range for the fine and coarse solvers definition, leading potentially to larger speed-up.

Thanks to numerical simulations in 1D and 3D we demonstrate the relevance of our parareal algorithm which uses two temporal discretizations and a different physical description in the fine and coarse solvers. We consider harmonic and anharmonic approximations of different potentials, namely the Lennard-Jones potential in the 1D case and the pair potential contribution of the Stillinger-Weber (SW) potential in the 3D case. Typically an harmonic approximation is used in the coarse solver while an harmonic/anharmonic approximation is taken for the fine resolution. Let us point out that this kind of approximation is not restrictive: this approximation is a very good one for atomic displacements close to the equilibrium distance and then can be applied to any pair potential. Moreover this approximation can be used for more complex potentials, such as the EAM (Embedded Atom Methods) potentials used for metals writing as the sum of a pair potential with an electronic contribution, or the whole SW potential which is the sum of a pair potential with a three-body angular contribution. By defining the coarse solver with the harmonic/anharmonic approximation of the pair potential of EAM or SW, we expect from our multi-step parareal algorithm to be able to treat various and complex MD configurations. From an efficiency point of view, the speed-up that we get using the parareal algorithm is all the more important as the ratio between the fine and the coarse resolution is large, provided that the coarse solver is not "too coarse". The quite attractive speed-up that we obtain for 3D silicon lattices - although the physical models of the fine and coarse solvers are quite close together - is a promising feature if one wants to apply our algorithm to more complex MD situations. Finally this parareal algorithm can also be viewed as a complementary algorithmic tool to be used in addition to DD techniques.

The paper is organized as follows. In Section 2 we recall the framework of MD, in particular the Verlet algorithm classically used for solving the equations of motion which are a high dimensional system of coupled ODE. Section 3 is devoted to algorithmic aspects of the temporal parallelization: first we recall in Section 3.1 the principle of the parareal algorithm which is an iterative process involving two levels of resolution, a coarse solver which is fast and run sequentially and a fine solver which is much more costly but performed in parallel. Then we detail in Section 3.2 our new parareal algorithm which allows us to use different temporal discretizations for the coarse/fine solvers, whereas the Verlet algorithm structure does not apply. The mathe-

matical properties of this multi-time step parareal algorithm are discussed in Section 3.3 showing its interesting behavior, namely the conservation of the total energy as well as the symplecticity. Eventually, numerical simulations are presented in Section 4: firstly in Section 4.1 the parareal algorithm is validated on a one-dimensional simplified test-case. In Section 4.2 we demonstrate by numerical means the efficiency of our strategy on a realistic three-dimensional periodical problem (diamond lattice of silicon atoms) by presenting simulations performed on parallel computers.

2. Framework: molecular dynamics. Molecular dynamics is devoted to describe the displacements of a great number of atoms. In a microcanonical system where the total energy, the volume and the number N_{at} of atoms are fixed, one has to solve the motion equations of classical mechanics

$$m_i \ddot{\mathbf{r}}_i = \mathbf{F}_i(\{\mathbf{r}_j\}),\ i = 1, \ldots, N_{at} \tag{2.1}$$

where $\mathbf{r}_i$ is the three-dimensional vector of atomic positions, m_i the mass and $\mathbf{F}_i = -\nabla_{\mathbf{r}_i} E_p$ the force deriving from the potential energy. By considering interactions of closest atomic neighbors in the potential energy

$$E_p = \frac{1}{2} \sum_{i \neq j, r_{ij} < r_c} V(r_{ij}) \tag{2.2}$$

where $r_{ij} = ||\mathbf{r}_i - \mathbf{r}_j||$ and r_c is a cutoff radius, we get the following coupled system of ODE

$$m_i \ddot{\mathbf{r}}_i = - \sum_{j \neq i, r_{ij} < r_c} V'(r_{ij}) \, \frac{\mathbf{r}_{ij}}{r_{ij}}. \tag{2.3}$$

There exist numerous interatomic potentials, such as the Lennard-Jones one (see [1] for more details). The choice of V is made according to what atomic system and what applications are under consideration. For a more convenient resolution of (2.3) the equations are adimensioned. As an illustration let us replace V by its harmonic/anharmonic approximation, as done in [20], by considering

$$\tilde{V}(r) = V(r_{min}) + \frac{V^{(2)}(r_{min})}{2}(r - r_{min})^2 + \frac{V^{(3)}(r_{min})}{6}(r - r_{min})^3 \tag{2.4}$$

where r_{min} is the equilibrium interatomic distance, that is to say $V^{(1)}(r_{min}) = 0$. Thanks to the variable changes $\mathbf{r}^* = \frac{\mathbf{r}}{a}$ and $t^* = \frac{t}{t_0}$ where a denotes the lattice constant and assuming that $m_i = m$, we get

$$\ddot{\mathbf{r}}_i^* = \mathbf{F}_i^* = - \sum_{j \neq i, r_{ij} < r_c} \left[(r_{ij}^* - r_{min}^*) + P_{anh}(r_{ij}^* - r_{min}^*)^2 \right] \frac{\mathbf{r}_{ij}}{r_{ij}} \tag{2.5}$$

with the anharmonic coefficient $P_{anh} = \frac{a}{2} \frac{V^{(3)}(r_{min})}{V^{(2)}(r_{min})}$ and with $t_0 = \sqrt{\frac{m}{V^{(2)}(r_{min})}}$. Let us emphasize that considering the approximation (2.4) instead of V is not restrictive because MD simulations involve small atomic displacements around r_{min} at low temperatures, meaning that the approximation (2.4) is a very good one for all pair potentials V. The low temperatures for which the harmonic approximation correctly describes the interatomic interactions are smaller than several tens of Kelvins, temperatures such as 300K being considered as high temperatures. On the contrary, the

temperature domain of validity of the harmonic/anharmonic approximation is not clearly defined. Nevertheless this approximation has been used successfully in [20] for silicon lattices at 300K, showing the robustness of this approximation at rather high temperatures.

Molecular dynamics simulations consist in solving non-dimensional equations such as (2.5). Typically t_0 is on the order of an atomic period of displacement ($\sim 10^{-12}$s); for example the whole integration time needed to get the equilibrium temperature is about ten atomic periods. From a computational point of view we refer the reader to [1] where different integrators are described (see also [21] for mathematical aspects). Algorithm 1 sums up the Verlet integrator that we will consider in our simulations.

Algorithm 1 Verlet algorithm

1: **for** $i = 1$ to N_{at} **do**
2: Compute the force $\mathbf{F}_i^*$.
3: Update the position with $\mathbf{r}_i^*(t+h) = 2\mathbf{r}_i^*(t) - \mathbf{r}_i^*(t-h) + h^2\mathbf{F}_i^*$.
4: **end for**

3. Temporal parallelization.

3.1. Principle. We describe here the parareal algorithm developed in [15] applied to the resolution of (2.5) on the interval $[0,T]$, $T > 0$. We denote by $\mathbf{R} = (\mathbf{r}_1^*, \ldots, \mathbf{r}_{N_{at}}^*)^T \in \mathbb{R}^{3N_{at}}$ the vector of all atomic positions of the system. Let us then assume that one has two levels of resolution: a coarse solver denoted by $\mathcal{G}$ and a fine one denoted by $\mathcal{F}$. The solver $\mathcal{G}$ is supposed to be very fast compared to $\mathcal{F}$; to set up ideas $\mathcal{G}$ integrates (2.5) using a simplified physics and/or using a simplified numerical scheme. By nature, $\mathcal{G}$ has to be run sequentially and $\mathcal{F}$ in parallel. To introduce parallelism the whole temporal interval is divided in N sub-intervals of length $h_N = \frac{T}{N}$; the parareal algorithm then consists in the following steps:

Algorithm 2 Parareal algorithm: principle

1: *Sequential* resolution on $[0,T]$: prediction of the trajectory with $\mathbf{R}_{n+1}^0 = \mathcal{G}(\mathbf{R}_n^0)$, $n = 0, \ldots, N-1$.
2: **for** $k = 0$ to k_{max} **do**
3: *Parallel* resolutions on $[t_n, t_{n+1}]$: compute $\mathcal{F}(\mathbf{R}_n^k)$, $n = 0, \ldots, N-1$.
4: *Sequential* corrections: $\mathbf{R}_{n+1}^{k+1} = \mathcal{G}(\mathbf{R}_n^{k+1}) + \mathcal{F}(\mathbf{R}_n^k) - \mathcal{G}(\mathbf{R}_n^k)$, $n = 0, \ldots, N-1$.
5: **end for**

In the previous algorithm $\mathbf{R}_n^k$ denotes $\mathbf{R}$ at the instant time $t_n = nh_N$ for the parareal iteration k. The correction steps are performed k_{max} times, until an acceptable precision is reached, typically the one provided by the fine solver. The great advantage of this strategy is the important speed-up that can be obtained. Indeed, if T_f (resp. T_c) denotes the CPU time needed by $\mathcal{F}$ (resp. $\mathcal{G}$) for the whole resolution of (2.5) on [0,T], the speed-up is given by

$$G_{CPU} = \frac{1}{\frac{k_{max}}{N} + (k_{max}+1)\frac{T_c}{T_f}}. \tag{3.1}$$

One can see from (3.1) that an equilibrium has to be found to get the optimal speed-up: if the ratio $\frac{T_c}{T_f}$ is too small - meaning that $\mathcal{G}$ is "too coarse" - too many corrections

steps will be needed, making G_{CPU} decreases. On the other side, if $\frac{T_c}{T_f}$ is close to 1, a very small number of corrections will be needed (let say $k_{max} = 1$) but the parallel computation ($\frac{T_f}{N} + 2T_c$) will be worst than the sequential one (T_f).

Lastly, assuming that $\mathcal{F}$ and $\mathcal{G}$ use different time-steps, the parareal algorithm can be seen as the replacement of a method of order m on the coarse temporal grid by a method of order $(k_{max} + 1)m$ on the same grid [16]. Another interpretation of this algorithm in term of multigrid and shooting method can also be found in [17].

3.2. A new parareal algorithm. In this section we detail how to build a *multi-time step* parareal algorithm for MD problems. We denote by δt the temporal discretization used for the fine solver $\mathcal{F}$ and by Δt the one taken for the coarse solver $\mathcal{G}$. The fine and the coarse propagation of the positions $\mathbf{R}_n$ on an interval of length h_N are respectively given by $\mathcal{F}(\mathbf{R}_n, h_N, \delta t)$ and $\mathcal{G}(\mathbf{R}_n, h_N, \Delta t)$.

The starting point of our methodology is the "classical" parareal algorithm used for one-dimensional MD problems. By classical we mean that $\mathcal{F}$ and $\mathcal{G}$ use different physical models with the *same temporal discretization*, namely $\Delta t = \delta t$. In the parareal algorithm based on a Verlet integrator, the initializations of the fine solver turns out to be the tricky point. This difficulty vanishes in the particular case where $\Delta t = \delta t$: indeed the initialization of the fine solver for the computation on the interval $[t_n, t_{n+1}]$ only requires the storage of the positions at times t_n and $t_n - \delta t$, since we have

$$\mathbf{R}^k(t_n + \delta t) = 2\mathbf{R}^k(t_n) - \mathbf{R}^k(t_n - \delta t) + \delta t^2 \mathbf{F}(\mathbf{R}^k(t_n)). \tag{3.2}$$

The positions at time $t_n - \delta t$ are available since the parareal positions are stored on the grid with time-step $\Delta t = \delta t$.

If we want to consider two different temporal discretizations, i.e. $\Delta t \neq \delta t$, we have to face a real difficulty. In this case, the positions $\mathbf{R}^k(t_n - \delta t)$ required for the fine solver initializations are unknown. As a matter of fact, the positions are only stored on the coarse grid since the parareal corrections are done on this temporal discretization. It would be also useless to store the positions given by the fine solver at times $t_n - \delta t$: it would introduce discontinuities since the parareal solution $\mathbf{R}_n^{k+1}$ and the fine one $\mathcal{F}(\mathbf{R}_{n-1}^k, h_N, \delta t)$ at time t_n differ from the quantity $\mathcal{G}(\mathbf{R}_{n-1}^{k+1}, h_N, \Delta t) - \mathcal{G}(\mathbf{R}_{n-1}^k, h_N, \Delta t)$. Consequently the extension of the classical parareal algorithm using a Verlet integrator to a multi-step version requires to be studied carefully.

We detail now our new algorithm. Since the positions $\mathbf{R}^k(t_n - \delta t)$ are unknown we estimate them thanks to the developments

$$\mathbf{R}^k(t_n - \delta t) \simeq \mathbf{R}^k(t_n) - \delta t \mathbf{V}^k(t_n) + \frac{\delta t^2}{2} \mathbf{F}(\mathbf{R}^k(t_n)) \tag{3.3}$$

where $\mathbf{V}^k \in \mathbb{R}^{3N_{at}}$ denotes the velocities of all the atoms at parareal iteration k. We need to have an estimate of $\mathbf{V}^k(t_n)$; for this we can integrate the equations $\dot{\mathbf{V}} = \mathbf{F}$ or just use the positions computed during the parareal algorithm. In both cases, the parareal algorithm has to make corrections on the positions and velocities at the same time, for computational efficiency purposes. In our case, since we only need velocities at instant times t_n it seems reasonable not to numerically solve equations on the velocities, but to build approximated velocities from the positions given by the fine and coarse solvers, and use them to make parareal corrections. In other words we choose the following updates

$$\mathbf{V}_n^{k+1} = \mathbf{V}_{\mathcal{G}}^{k+1}(t_n) + \mathbf{V}_{\mathcal{F}}^k(t_n) - \mathbf{V}_{\mathcal{G}}^k(t_n) \tag{3.4}$$

where $\mathbf{V}_{\mathcal{G}}^k(t_n)$ (resp. $\mathbf{V}_{\mathcal{F}}^k(t_n)$) denotes the coarse (resp. fine) velocities at time t_n and parareal iteration k, and $\mathbf{V}_n^{k+1}$ are the corrected velocities at time t_n. To take into account the velocities dynamics, we use the following second-order developments

$$\mathbf{V}_{\mathcal{G}}^k(t_n) = \frac{\mathbf{R}^k(t_n) - \mathbf{R}^k(t_n - \Delta t)}{\Delta t} + \frac{\Delta t}{2}\mathbf{F}_{\mathcal{G}}(\mathbf{R}^k(t_n)) \tag{3.5}$$

and

$$\mathbf{V}_{\mathcal{F}}^k(t_n) = \frac{\mathbf{R}^k(t_n) - \mathbf{R}^k(t_n - \delta t)}{\delta t} + \frac{\delta t}{2}\mathbf{F}_{\mathcal{F}}(\mathbf{R}^k(t_n)), \tag{3.6}$$

where $\mathbf{F}_{\mathcal{G}}(\mathbf{R})$ (resp. $\mathbf{F}_{\mathcal{F}}(\mathbf{R})$) denotes the forces computed with the coarse (resp. fine) solver. Indeed, by remarking that we have for example

$$\frac{\mathbf{R}^k(t_n) - \mathbf{R}^k(t_n - \Delta t)}{\Delta t} \simeq \mathbf{V}_{\mathcal{G}}^k(t_n - \frac{\Delta t}{2}),$$

we can rewrite (3.5) as

$$\mathbf{F}_G^k(t_n) = \frac{\mathbf{V}_G^k(t_n) - \mathbf{V}_G^k(t_n - \frac{\Delta t}{2})}{\frac{\Delta t}{2}}$$

which is the discretization of the continuous equation $\dot{\mathbf{V}} = \mathbf{F}$. Lastly, one can notice that the differences $\mathbf{V}_{\mathcal{F}}^k(t_n) - \mathbf{V}_{\mathcal{G}}^k(t_n)$ can be performed in parallel in each interval while the computation of the coarse velocities $\mathbf{V}_{\mathcal{G}}^{k+1}(t_n)$ is sequential, since it requires the knowledge of the updated positions $\mathbf{R}^{k+1}$ from the relations

$$\mathbf{V}_{\mathcal{G}}^{k+1}(t_n) = \frac{\mathbf{R}^{k+1}(t_n) - \mathbf{R}^{k+1}(t_n - \Delta t)}{\Delta t} + \frac{\Delta t}{2}\mathbf{F}_{\mathcal{G}}(\mathbf{R}^{k+1}(t_n)). \tag{3.7}$$

To summarize, we initialize the fine solver on each interval $[t_n, t_{n+1}]$ thanks to (3.3), where the velocities $\mathbf{V}^k(t_n)$ are updated with parareal corrections (3.4) involving coarse and fine velocities numerically computed from the positions. We can also remark that rewriting (3.6) leads to (3.3), meaning that second-order approximations on the velocities implies second-order approximations on the positions required for the fine solver initializations. This choice was not obvious at first sight, and leads to correct numerical solutions as it will be shown in Section 4.

Concerning the initializations of the coarse solver, we proceed as in the classical algorithm: parareal updates are done on the positions at times $t_n - \Delta t$ ($1 \leq n \leq N-1$) thanks to the following corrections

$$\mathbf{R}^{k+1}(t_{n+1} - \Delta t) = \mathcal{G}(\mathbf{R}_n^{k+1}, h_N - \Delta t, \Delta t) + \mathcal{F}(\mathbf{R}_n^k, h_N - \Delta t, \delta t) - \mathcal{G}(\mathbf{R}_n^k, h_N - \Delta t, \Delta t). \tag{3.8}$$

For the sake of readability, we sum-up the different steps of the multi-time step parareal method in algorithm 3.

3.3. Mathematical properties. In this section we give mathematical properties satisfied by the coarse and fine solvers used in our multi-time step parareal algorithm. In that aim, we need to introduce the Störmer-Verlet (SV) integrator [21] with which the coarse and fine solvers will be compared, and recall its properties.

Algorithm 3 Multi-time step parareal algorithm for MD

1: *Sequential* resolution on $[0,T]$: prediction of the trajectory with $\mathbf{R}^0_{n+1} = \mathcal{G}(\mathbf{R}^0_n, h_N, \Delta t)$ and $\mathbf{R}^0(t_{n+1}-\Delta t) = \mathcal{G}(\mathbf{R}^0_n, h_N - \Delta t, \Delta t),\ 0 \le n \le N-1$. Storage of velocities $\mathbf{V}^0_n,\ 1 \le n \le N-1$.

2: **for** $k = 0$ to k_{max} **do**

3: *Parallel* resolutions on $[t_n, t_{n+1}]$: compute $\mathcal{F}(\mathbf{R}^k_n, h_N, \delta t),\ 0 \le n \le N-1$. Initializations with

$$\mathbf{R}^k(t_n - \delta t) = \mathbf{R}^k(t_n) - \delta t \mathbf{V}^k(t_n) + \frac{\delta t^2}{2}\mathbf{F}_{\mathcal{F}}(\mathbf{R}^k(t_n)).$$

4: *Sequential* corrections at times t_n:

$$\mathbf{R}^{k+1}_{n+1} = \mathcal{G}(\mathbf{R}^{k+1}_n, h_N, \Delta t) + \mathcal{F}(\mathbf{R}^k_n, h_N, \delta t) - \mathcal{G}(\mathbf{R}^k_n, h_N, \Delta t),\ 0 \le n \le N-1.$$

Sequential corrections at times $t_n - \Delta t$ (for coarse initializations):

$$\begin{aligned}\mathbf{R}^{k+1}(t_{n+1} - \Delta t) = {} & \mathcal{G}(\mathbf{R}^{k+1}_n, h_N - \Delta t, \Delta t) \\ & + \mathcal{F}(\mathbf{R}^k_n, h_N - \Delta t, \delta t) - \mathcal{G}(\mathbf{R}^k_n, h_N - \Delta t, \Delta t),\ 0 \le n \le N-1.\end{aligned}$$

5: Update of velocities from positions:

$$\mathbf{V}^{k+1}_n = \mathbf{V}^{k+1}_{\mathcal{G}}(t_n) + \mathbf{V}^k_{\mathcal{F}}(t_n) - \mathbf{V}^k_{\mathcal{G}}(t_n),\ 1 \le n \le N-1$$

where $\mathbf{V}^k_{\mathcal{G}}$ and $\mathbf{V}^k_{\mathcal{F}}$ are computed thanks to (3.5) and (3.6) and with

$$\mathbf{V}^{k+1}_{\mathcal{G}}(t_n) = \frac{\mathbf{R}^{k+1}_n - \mathbf{R}^{k+1}(t_n - \Delta t)}{\Delta t} + \frac{\Delta t}{2}\mathbf{F}_{\mathcal{G}}(\mathbf{R}^{k+1}_n).$$

6: **end for**

The SV algorithm is usually formulated in an Hamiltonian framework: the total energy is written as

$$H(\mathbf{R}, \mathbf{P}) = E_c(\mathbf{P}) + E_p(\mathbf{R})$$

where $E_p(\mathbf{R})$ is the potential energy and $E_c(\mathbf{P}) = \frac{1}{2}\mathbf{P}^T M^{-1}\mathbf{P}$ the kinetic energy with the impulsion $\mathbf{P} \in \mathbb{R}^{3N_{at}}$ defined by $\mathbf{P} = M\dot{\mathbf{R}}$. The SV algorithm is then given by the non-dimensional equations

$$\left\{\begin{array}{l}\mathbf{P}(t + \frac{h}{2}) = \mathbf{P}(t) - \frac{h}{2}\nabla E_p(\mathbf{R}(t)), \\ \mathbf{R}(t+h) = \mathbf{R}(t) + h\nabla E_c(\mathbf{P}(t + \frac{h}{2})) = \mathbf{R}(t) + hM^{-1}\mathbf{P}(t + \frac{h}{2}), \\ \mathbf{P}(t+h) = \mathbf{P}(t + \frac{h}{2}) - \frac{h}{2}\nabla E_p(\mathbf{R}(t+h)).\end{array}\right. \tag{3.9}$$

The SV algorithm is symplectic as a composition of two symplectic flows, as shown in [22]. Moreover the SV algorithm is of order 2 since it can be seen as a partitioned Runge-Kutta method (see [21]). Let us note that we can directly prove this result as shown in Proposition 3.1.

PROPOSITION 3.1. *The SV algorithm (3.9) preserves the total energy up to the second order, namely*

$$H(\mathbf{P}^{n+1},\mathbf{R}^{n+1}) = H(\mathbf{P}^{n},\mathbf{R}^{n}) + \mathcal{O}(h^2). \tag{3.10}$$

Proof. From the equations (3.9) we get

$$\mathbf{P}^{n+1} = \mathbf{P}^{n} - h\nabla E_p(\mathbf{R}^n) + \mathcal{O}(h^2) \tag{3.11}$$

which leads for the kinetic contribution to the relation

$$\begin{aligned} E_c(\mathbf{P}^{n+1}) &= \frac{1}{2}\sum_{i=1}^{N_{at}} m_i^{-1}||\mathbf{p}_i^n - h\nabla_{\mathbf{r}_i}E_p(\mathbf{R}^n)||^2 + \mathcal{O}(h^2) \\ &= E_c(\mathbf{P}^n) - h\sum_{i=1}^{N_{at}} m_i^{-1}\big(\mathbf{p}_i^n, \nabla_{\mathbf{r}_i}E_p(\mathbf{R}^n)\big) + \mathcal{O}(h^2). \end{aligned} \tag{3.12}$$

For the potential energy we need to develop $E_p(\mathbf{R}^{n+1}) = \dfrac{1}{2}\displaystyle\sum_{i\neq j} V\big(||\mathbf{r}_i^{n+1} - \mathbf{r}_j^{n+1}||\big)$ up to the second order. With the two first relations of (3.9) we have

$$\mathbf{r}_i^{n+1} = \mathbf{r}_i^n + hm_i^{-1}\mathbf{p}_i^n + \mathcal{O}(h^2),$$

so that

$$||\mathbf{r}_i^{n+1} - \mathbf{r}_j^{n+1}|| = ||\mathbf{r}_i^n - \mathbf{r}_j^n||\left(1 + h\big(\frac{\mathbf{r}_i^n - \mathbf{r}_j^n}{||\mathbf{r}_i^n - \mathbf{r}_j^n||^2}, m_i^{-1}\mathbf{p}_i^n - m_j^{-1}\mathbf{p}_j^n\big)\right) + \mathcal{O}(h^2).$$

This leads to

$$\begin{aligned} V\big(||\mathbf{r}_i^{n+1} - \mathbf{r}_j^{n+1}||\big) &= V\big(||\mathbf{r}_i^n - \mathbf{r}_j^n||\big) \\ &\quad + h\left(\frac{\mathbf{r}_i^n - \mathbf{r}_j^n}{||\mathbf{r}_i^n - \mathbf{r}_j^n||}, m_i^{-1}\mathbf{p}_i^n - m_j^{-1}\mathbf{p}_j^n\right)V'(||\mathbf{r}_i^n - \mathbf{r}_j^n||) + \mathcal{O}(h^2) \end{aligned}$$

and then we get

$$E_p(\mathbf{R}^{n+1}) = E_p(\mathbf{R}^n) + \frac{h}{2}\sum_{i\neq j}\left(\frac{\mathbf{r}_i^n - \mathbf{r}_j^n}{||\mathbf{r}_i^n - \mathbf{r}_j^n||}, m_i^{-1}\mathbf{p}_i^n - m_j^{-1}\mathbf{p}_j^n\right)V'(||\mathbf{r}_i^n - \mathbf{r}_j^n||) + \mathcal{O}(h^2). \tag{3.13}$$

To get the expected result we have to combine (3.12) and (3.13). Remarking that

$$\nabla_{\mathbf{r}_i}E_p(\mathbf{R}^n) = 2\times\frac{1}{2}\sum_{j\neq i} V'(||\mathbf{r}_i^n - \mathbf{r}_j^n||)\frac{\mathbf{r}_i^n - \mathbf{r}_j^n}{||\mathbf{r}_i^n - \mathbf{r}_j^n||}$$

we rewrite (3.12) as

$$E_c(\mathbf{P}^{n+1}) = E_c(\mathbf{P}^n) - h\sum_{j\neq i} m_i^{-1}\left(\mathbf{p}_i^n, \frac{\mathbf{r}_i^n - \mathbf{r}_j^n}{||\mathbf{r}_i^n - \mathbf{r}_j^n||}\right)V'(||\mathbf{r}_i^n - \mathbf{r}_j^n||) + \mathcal{O}(h^2). \tag{3.14}$$

By adding (3.13) and (3.14) the terms in h vanish and we finally get

$$E_c(\mathbf{P}^{n+1}) + E_p(\mathbf{R}^{n+1}) = E_c(\mathbf{P}^n) + E_p(\mathbf{R}^n) + \mathcal{O}(h^2).$$

□

PROPOSITION 3.2. *The fine and coarse solvers used in the parareal algorithm 3 preserve the total energy up to the second order and are symplectic integrators.*

Proof. Firstly, we show that the fine solver used in algorithm 3 is a reformulation of the SV algorithm. Writing the first two equations of (3.9) in terms of positions and velocities at $t = t_n$ and with $h = \delta t$, and using $-\nabla E_p = \mathbf{F}$, leads to

$$\begin{cases} \mathbf{V}(t_n + \frac{\delta t}{2}) = \mathbf{V}(t_n) + \frac{\delta t}{2} M^{-1}\mathbf{F}(\mathbf{R}(t_n)), \\ \mathbf{R}(t_n + \delta t) = \mathbf{R}(t_n) + \delta t \mathbf{V}(t_n + \frac{\delta t}{2}). \end{cases} \tag{3.15}$$

Then we deduce the velocities at t_n by the relation

$$\mathbf{V}(t_n) = \frac{\mathbf{R}(t_n + \delta t) - \mathbf{R}(t_n)}{\delta t} - \frac{\delta t}{2} M^{-1}\mathbf{F}(\mathbf{R}(t_n))$$

which is nothing else but equation (3.6) written between t_n and $t_n + \delta t$ instead of $t_n - \delta t$ and t_n. Therefore the fine solver $\mathcal{F}$ used in algorithm 3 is equivalent to the SV algorithm, which is symplectic and of order 2 for the conservation of energy, namely the local error is in $\mathcal{O}(\delta t^2)$.

Concerning the coarse solver $\mathcal{G}$ based on

$$\mathbf{R}(t_n + \Delta t) = 2\mathbf{R}(t_n) - \mathbf{R}(t_n - \Delta t) + \Delta t^2 \mathbf{F}(\mathbf{R}(t_n))$$

where $\mathbf{F}$ are the non-dimensional forces, we show this relation is equivalent to the SV algorithm up to $\mathcal{O}(\Delta t^3)$. Indeed, rewriting the second SV equation at $t = t_n$ and with $h = \Delta t$ leads to

$$\mathbf{R}(t_n + \Delta t) = \mathbf{R}(t_n) + \Delta t M^{-1}\left(\mathbf{P}(t_n - \frac{\Delta t}{2}) - \Delta t \nabla E_p(\mathbf{R}(t))\right).$$

Since $\mathbf{P}(t_n - \frac{\Delta t}{2}) = M\dot{\mathbf{R}}(t_n - \frac{\Delta t}{2})$ and $\frac{\mathbf{R}(t_n) - \mathbf{R}(t_n - \Delta t)}{\Delta t} = \dot{\mathbf{R}}(t_n - \frac{\Delta t}{2}) + \mathcal{O}(\Delta t^2)$ we get

$$\begin{aligned} \mathbf{R}(t_n + \Delta t) &= \mathbf{R}(t_n) + \Delta t M^{-1}\left(M\frac{\mathbf{R}(t_n) - \mathbf{R}(t_n - \Delta t)}{\Delta t} + \Delta t \mathbf{F}(\mathbf{R}(t_n)) + \mathcal{O}(\Delta t^2)\right) \\ &= 2\mathbf{R}(t_n) - \mathbf{R}(t_n - \Delta t) + \Delta t^2 M^{-1}\mathbf{F}(\mathbf{R}(t_n)) + \mathcal{O}(\Delta t^3). \end{aligned} \tag{3.16}$$

Since the SV algorithm is of order 2 we deduce that the coarse solver $\mathcal{G}$ is also of order 2, meaning the local error made on the total energy is in $\mathcal{O}(\Delta t^2)$.
□

COROLLARY 3.3. *The parareal algorithm 3 preserves the total energy during time, is of order 2 and is a symplectic integrator.*

Proof. At convergence of algorithm 3, i.e. when k becomes large, the parareal solution $\mathbf{R}^k$ converges toward the fine solution (at $k = 0$ the parareal solution is equal to the coarse solution). Therefore the converged parareal solution inherits from all properties of the fine solver (see Proposition 3.2).
□

4. Numerical results.

4.1. Validation: 1D test-case: the Lennard-Jones lattice. For a first validation of the multi-time step parareal algorithm 3, we consider a simplified one-dimensional problem, where the Lennard-Jones potential

$$V_{LJ}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{4.1}$$

is replaced by its harmonic/anharmonic approximation (see (2.4)). By considering a cut-off radius to the first neighbor in (4.1), each atom i interacts with its two closest neighbors $i \pm 1$ and the non-dimensional motion equations for displacements write

$$\ddot{u}_i^* = (u_{i+1}^* + u_{i-1}^* - 2u_i^*) + P_{anh}\left((u_{i+1}^* - u_i^*)^2 - (u_{i-1}^* - u_i^*)^2\right) \tag{4.2}$$

with $P_{anh} = \frac{V_{LJ}^{(3)}(r_{min})}{2V_{LJ}^{(2)}(r_{min})} \simeq -2.6727$ and $r_{min} = \sigma 2^{1/6}$, the variable change being defined by $u^* = \frac{u}{r_{min}}$. We choose "frozen" boundary conditions, meaning that $u_{-1} = u_{N_at+1} = 0$. In our simulations the positions are initialized considering random perturbations of the Bravais lattice with zero velocities, the perturbation being on the order of 1% of r_{min}^*. The parameters chosen for the simulation are defined in Table 4.1; the anharmonic coefficient P_{anh} is taken 10 times larger than the physical value so that the dynamics of the coarsely predicted solution and the reference one are different enough, to make more readable the parareal convergence study.

T	N_{at}	P_{anh}	N	$\mathcal{F}$	$\mathcal{G}$	Δt	δt	k_{max}
20	50	-20	10	harmonic+anharmonic	harmonic	0.2	0.02	N

TABLE 4.1
Parameters chosen for the one-dimensional test-case.

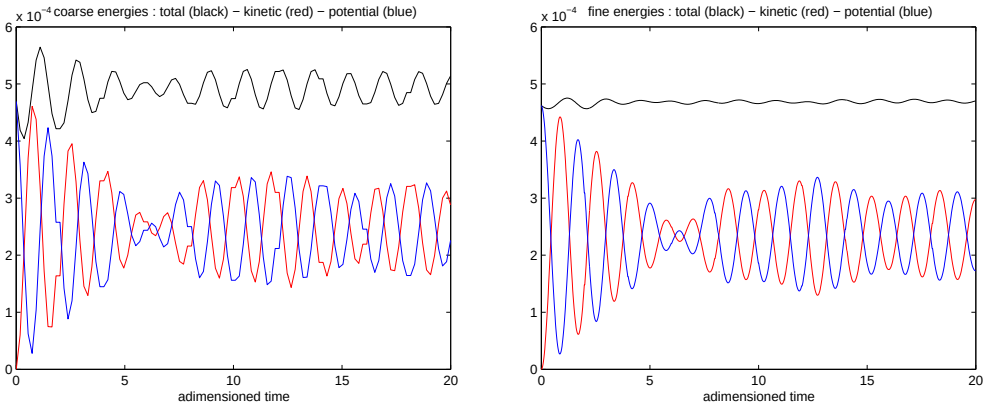

FIG. 4.1. *4.1.a: historic of the energies during the prediction step. 4.1.b: historic of the energies obtained with the fine solver.*

As shown in Fig. 4.1 the parareal algorithm preserves the total energy $E_c^* + E_p^*$

where the non-dimensional kinetic and potential energies are defined by

$$E_c^* = \frac{1}{2}\sum_{i=1}^{N_{at}} v_i^{*2}$$

$$E_p^* = \frac{1}{2}\sum_{i=1}^{N_{at}} \frac{1}{2}\Big((u_{i+1}^* - u_i^*)^2 + (u_{i-1}^* - u_i^*)^2\Big) + \frac{P_{anh}}{3}\Big((u_{i+1}^* - u_i^*)^3 - (u_{i-1}^* - u_i^*)^3\Big).$$

In Fig. 4.1 one can observe small oscillations for the total energy corresponding to the second-order precision of the Verlet algorithm. In the prediction step, the maximal amplitude of the total energy is about 0.17% and of about 0.02% in the fine resolution. First of all these results are coherent with the error which is generally allowed on the total energy in MD, namely 0.05%. Moreover the maximal amplitude of the fine energy is about ten times smaller than the coarse one, which corresponds to the ratio $\frac{\Delta t}{\delta t}$. This behavior is in agreement with Proposition 3.2 showing that the coarse solver $\mathcal{G}$ locally preserves the energy in $\mathcal{O}(\Delta t^2)$ while the fine solver $\mathcal{F}$ do the same in $\mathcal{O}(\delta t^2)$. Therefore the global error made on the energy is in $\mathcal{O}(\Delta t)$ for $\mathcal{G}$ and $\mathcal{O}(\delta t)$ for $\mathcal{F}$.

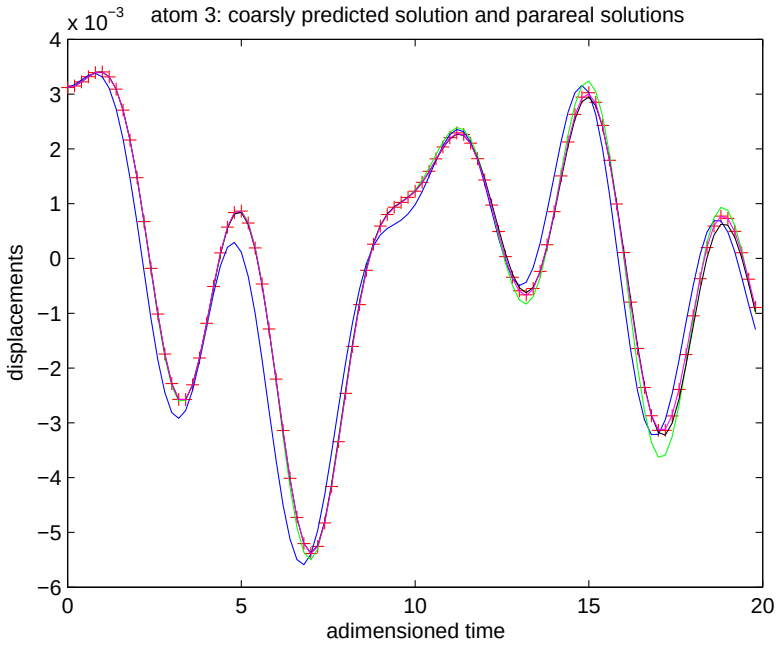

FIG. 4.2. *Reference solution (in red with plus sign), predicted solution (in blue) and three corrected parareal solutions (in green for $k = 1$, in black for $k = 2$, in magenta for $k = 3$).*

In Fig. 4.2 we represent for a selected atom the dynamics of the reference solution computed with the fine solver, the solution predicted coarsely and several parareal solutions. One can see the quick convergence of the parareal solutions toward the reference solution: the parareal solution with three corrections is practically indistinguishable from the reference one. The convergence of the parareal solutions toward the reference trajectory $\mathbf{R}^{ref}$ is enhanced in Fig. 4.3, where the historic of the errors

$\log_{10}|\mathbf{R}^k - \mathbf{R}^{ref}|$ are plotted for different values of k. Concerning the algorithmic convergence we also plot in Fig. 4.4 the time evolution of the Newton increment at each interface $t_n = nh_N$, $2 \leq n \leq N$, the increment at time t_n being defined by

$$r_n^k = ||\mathcal{F}(\mathbf{R}_{n-1}^k, h_N, \delta t) - \mathbf{R}_n^{k+1}||_{\mathbb{R}^{N_{at}}}. \tag{4.3}$$

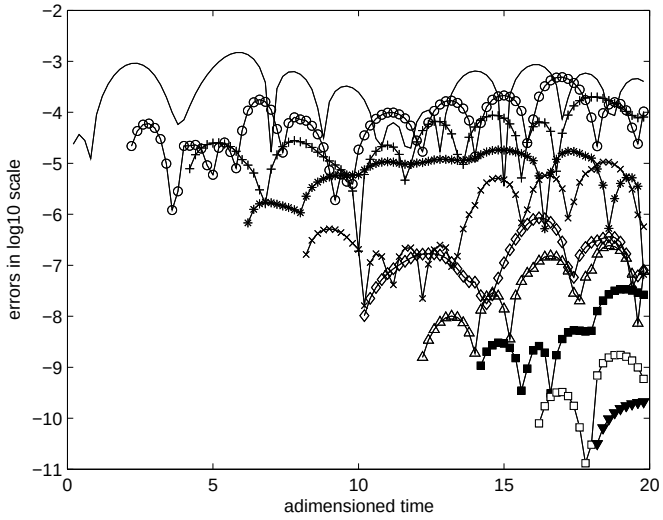

FIG. 4.3. *Historic of the errors in* $\log_{10}$ *scale between parareal solutions and the reference one, for different values of* k*: circles for* $k = 1$*, plus sign for* $k = 2$*, asterisks for* $k = 3$*, cross for* $k = 4$*, diamonds for* $k = 5$*, white triangles for* $k = 6$*, black squares for* $k = 7$*, white squares for* $k = 8$*, black triangles for* $k = 9$*. The error between the predicted solution and the reference one is in solid line.*

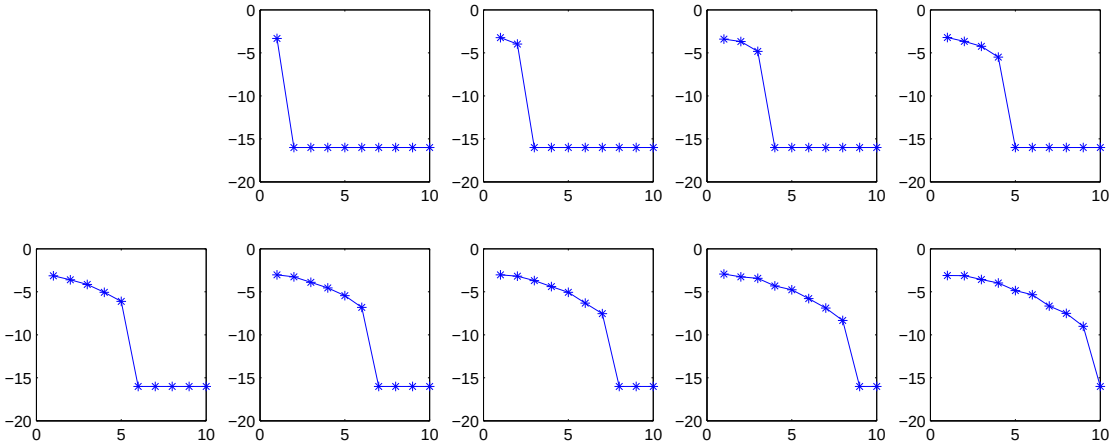

FIG. 4.4. *Historic in* $\log_{10}$ *scale of the increments (4.3) at the instant times* t_n*,* $2 \leq n \leq N$*.*

The expected behavior of the parareal algorithm is numerically checked, namely the increment at time t_n reaches the computer precision (10^{-16}) in at most n iterations. Let us finally mention that wet get an atomic displacement period T_r of about 2 times larger than the kinetic energy period T_{E_c}, as seen in Figs. 4.1 and 4.2. Indeed equations (4.2) taken with $P_{anh} = 0$ are nothing else but an harmonic oscillator for which $T_r = 2T_{E_c} = \pi$.

4.2. 3D test-case: silicon diamond lattice. Now we apply the multi-time parareal algorithm to a more realistic problem to study its computational efficiency. For this, let us consider the diamond lattice of silicon atoms, which is made of two face-centered-cubic (fcc) lattices. The second fcc lattice is deduced from the first one by a translation of $(\frac{a}{4}, \frac{a}{4}, \frac{a}{4})$, where a is the lattice constant. Each unit cell contains 8 atoms and this pattern is duplicated in the 3 directions to get a periodical box $[0, L]^3$ as shown in Fig. 4.5. Each atom has 4 neighbors, the closest interatomic distance being defined by $r_{min} = \frac{a\sqrt{3}}{4}$. Finally, the motion equations are given by (2.5) with $P_{anh} \simeq -3.8798$, which are now coupled ODE. The value of the anharmonic coefficient P_{anh} is obtained by fitting the bulk modulus coefficient and the Grüneisen parameter, see [20].

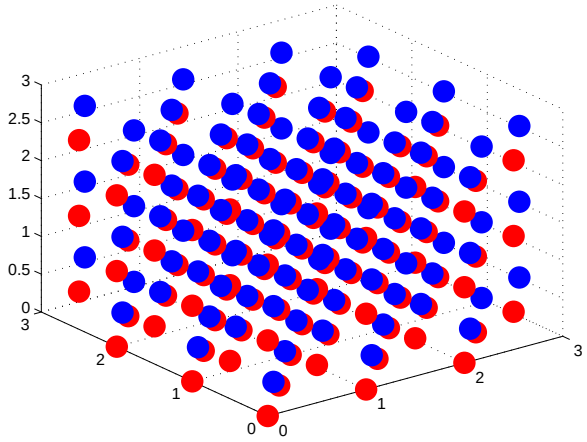

FIG. 4.5. *Diamond lattice with 216 silicon atoms corresponding to 27 unit cells: fcc lattice (in red) and shifted fcc lattice (in blue).*

As before we initialize the atomic positions with small random perturbations of about 1% of r^*_{min} of the Bravais lattice with zero velocities. From an algorithmic point of view, the periodic boundary conditions require to treat carefully the differences between the fine and the coarse atomic positions that are iteratively computed. Indeed, since the solvers $\mathcal{F}$ and $\mathcal{G}$ are different - different potentials to compute the forces and different time steps are used - the fine and coarse positions that we get can be very different for atoms close to the boundary of the box. Consequently we need to detect the cases where the differences $\mathbf{R}^k_{\mathcal{F}}(t_n) - \mathbf{R}^k_{\mathcal{G}}(t_n)$ are large to take into account the periodic boundary conditions. For these atoms we compute as for the forces calculations the minimal distance $||\mathbf{R}^k_{\mathcal{F}}(t_n) - I(\mathbf{R}^k_{\mathcal{G}}(t_n))||$ where $I(\mathbf{R})$ denotes the images of $\mathbf{R}$, namely $\{\mathbf{R} + (0 \pm L, 0 \pm L, 0 \pm L)\}$.

In order to get speed-up results, two studies are presented: at first δt and Δt are fixed while N is variable, secondly N is fixed and the ratio $\frac{\Delta t}{\delta t}$ is variable. In both test-cases the parareal algorithm is stopped at iteration k when the global increment is small enough, i.e.

$$r^k = \left(\frac{1}{N} \sum_{n=1}^{N} (r_n^k)^2 \right)^{1/2} = \left(\frac{1}{N} \sum_{n=1}^{N} ||\mathcal{F}(\mathbf{R}_{n-1}^k, h_N, \delta t) - \mathbf{R}_n^{k+1}||_{\mathbb{R}^{N_{at}}}^2 \right)^{1/2} < \varepsilon. \quad (4.4)$$

The threshold ε is chosen so that the previous global increment represents a small fraction of the typical atomic displacements which depend on the order of magnitude of the initial perturbations. Moreover for each simulation the number N_{procs} of processors is equal to the number N of sub-intervals. The simulations corresponding to the results that are presented were performed on a cluster made of 32 nodes with 2 processors AMD Opteron 64 bits dual core with speed 2.4 GHz, the 128 cores being connected by an infinyband gigabit network. The numerical code is written in Fortran 90 and uses the MPI library for the parallel features.

1. δt and Δt fixed, N variable. The simulation parameters are given in Tab. 4.2. As shown in Tab. 4.3 where the sequential CPU time $T_{seq} = T_f$ is compared to the parareal CPU time $T_{par} = k_{max} \frac{T_f}{N} + (k_{max} + 1) T_c$, we get interesting speed-up when N increases, with factors of more than 10.

T	N_{at}	P_{anh}	$\mathcal{F}$	$\mathcal{G}$	Δt	δt	ϵ
20	21952	-3.8798	harmonic+anharmonic	harmonic	0.05	10^{-4}	10^{-4}

TABLE 4.2
Parameters taken for the 3D efficiency tests (N variable).

N	T_{seq}	T_{par}	G_{CPU}	k_{max}
10	3404	871	3.91	4
20	3412	818	4.17	4
50	3402	479	7.10	4
100	3449	311	11.09	4

TABLE 4.3
Speed-up and CPU times (seconds) in the 3D test-case as a function of N.

2. N fixed, $\frac{\Delta t}{\delta t}$ variable. We consider here a fixed number of processors and a fixed fine time-step δt in order to make coherent the comparisons; then we take different values for Δt. The simulations parameters are defined in Tab. 4.4 and the results presented in Tab. 4.5. As shown in Fig. 4.6, for a fixed number of processors we get an optimal speed-up for a particular ratio $\frac{\Delta t}{\delta t}$ (of about 100 in this case). This example illustrates the equilibrium to be found between a small number of parareal iterations and an attractive ratio $\frac{\Delta t}{\delta t}$, as mentioned in Section 3. When $\frac{\Delta t}{\delta t}$ is too high - meaning that the solver $\mathcal{G}$ is "too coarse" - the number of parareal iterations k_{max} needed to reach the convergence increases. On the opposite, when $\frac{\Delta t}{\delta t}$ is too small - meaning that $\mathcal{G}$ is too close to $\mathcal{F}$ - the ratio $\frac{T_c}{T_f}$ becomes too large and the speed-up decreases (see (3.1)).

T	N_{at}	P_{anh}	$\mathcal{F}$	$\mathcal{G}$	N	δt	ϵ
20	21952	-3.8798	harmonic+anharmonic	harmonic	20	10^{-4}	10^{-4}

TABLE 4.4
Parameters taken for the 3D efficiency tests with N fixed.

$\frac{\Delta t}{\delta t}$	10	20	50	100	200	500	1000	2000
T_{seq}	3363	3399	3357	3394	3433	3412	3404	3409
T_{par}	1109	574	520	434	616	818	983	1189
G_{CPU}	3.03	5.92	6.45	7.82	5.57	4.17	3.46	2.87
k_{max}	2	2	2	2	3	4	5	8

TABLE 4.5
Speed-up and CPU times (seconds) in the 3D test-case with N fixed.

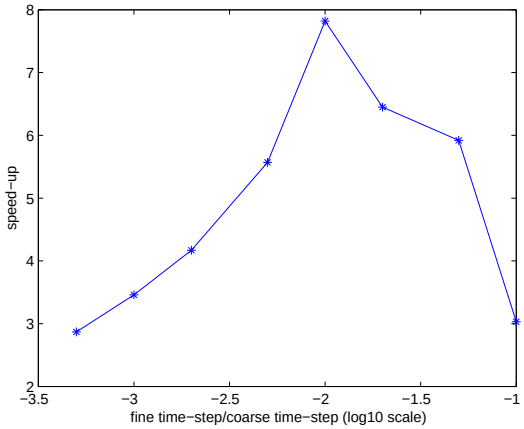

FIG. 4.6. *Graphical representation of Tab. 4.5:* $G_{CPU} = f\left(\log_{10}(\frac{\delta t}{\Delta t})\right)$.

Finally as an illustration we give numerical results associated to the parameters taken in Tab. 4.2 with $N = 100$. In Fig. 4.7 we represent the solution predicted coarsely, the reference solution and several parareal trajectories for a selected atom. Again we get a satisfying behavior of the parareal algorithm: the parareal solution with three corrections completely fits to the reference trajectory. In Fig. 4.8 we plot the instantaneous temperature T obtained from the non-dimensional kinetic energy E_c^* associated to the converged parareal solution, namely

$$T(t) = \frac{a^2\beta}{\frac{3}{2}N_{at}k_B}\, E_c^*(t) \tag{4.5}$$

where $a = 5.45\,10^{-10}$ m is the lattice constant, $\beta = 50.0614\,\mathrm{kg\,s^{-2}}$ is the second derivative of the potential at r_{min} and $k_B = 1,3806\,10^{-23}$ J K^{-1} denotes the Bolzmann constant.

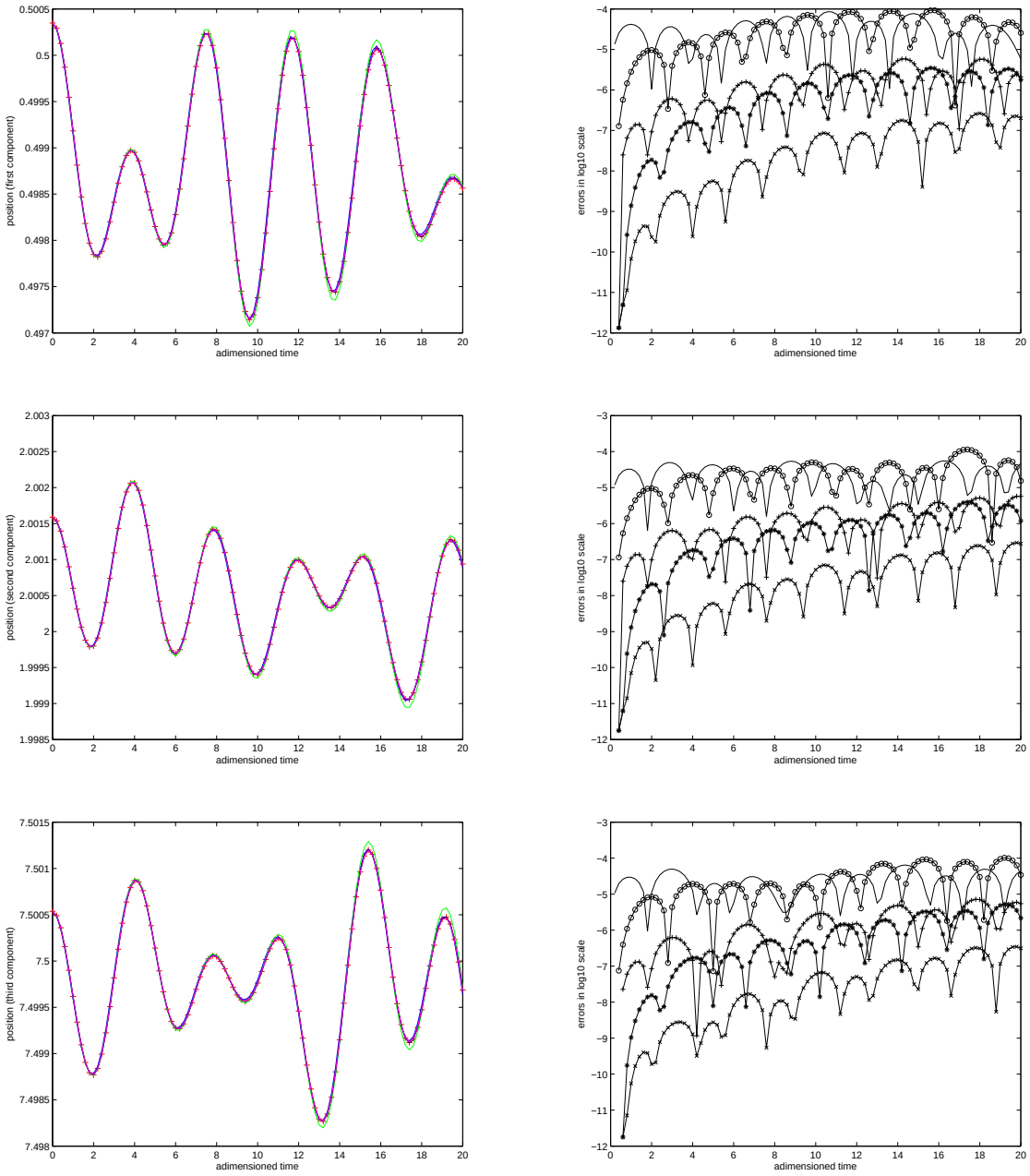

FIG. 4.7. *3D test-case with parameters of Tab. 4.2 for* $N = 100$. *Left side: for a selected atom, three components of the reference solution (in red with sign plus), solution predicted coarsely (in blue) and parareal trajectories (in green for* $k = 1$, *in black for* $k = 2$, *in magenta for* $k = 3$*). Right side: historic of the corresponding errors in* $\log_{10}$ *scale between parareal solutions and the reference one, for different values of* k*: circles for* $k = 1$, *plus sign for* $k = 2$, *asterisks for* $k = 3$, *cross for* $k = 4$. *The error between the predicted solution and the reference one is in solid line.*

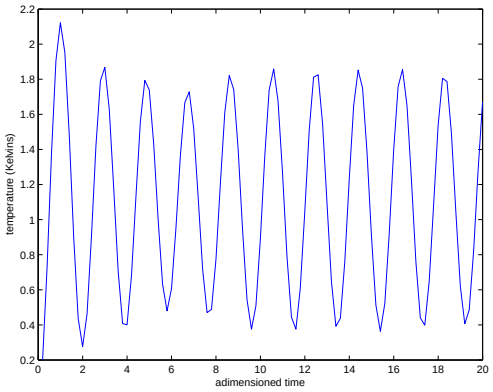

FIG. 4.8. *Historic of the instantaneous temperature (4.5) computed from the converged parareal solution. The mean temperature is about 1.114 Kelvins.*

5. Conclusion. We have built a new parareal algorithm based on a conservative, symplectic integrator of order two, since the Verlet algorithms that we use can be reformulated as Störmer-Verlet integrators. Even if the velocities are not numerically integrated they are useful to initialize the fine parallel resolutions, their computation being performed thanks to specific parareal updates. In that sense, our parareal algorithm seems to be an original contribution for MD applications. Moreover, the multi-time stepping is also an interesting feature which allows us to get attractive speed-up, as shown in the three-dimensional simulations where speed-up of more than 10 has been obtained for lattices of 21952 atoms. The impact should be significative in the MD framework when using this parareal algorithm on much more complex MD potentials, and by combining it to decomposition domain techniques. According to what MD application is under consideration, one has to correctly define the temporal discretization of the coarse solver in order to get the optimal speed-up.

Acknowledgments. This work is supported by the French national network research program Agence Nationale de la Recherche ANR, project PITAC CIS 2006 (*Parallélisation Incluant le Temps pour Accélérer les Calculs*, Scientific Coordinator of the project Prof. Y. Maday). The authors would like to thank M. Duarte and D. Alviso, two students who did a research project at EM2C laboratory in the framework of the Master of Science Energy and Aerospace Engineering at Ecole Centrale Paris, and who initiated the work on the parareal algorithms applied to molecular dynamics. The help of Stéphane de Chaisemartin, Ph.D student of EM2C laboratory, who designed the structure of the Fortran code dedicated to the parareal algorithm on parallel architecture with MPI library, is gratefully acknowledged.

REFERENCES

[1] D.C. Rapaport, *The Art of Molecular Dynamics Simulation*, Cambridge University Press, 1995.
[2] S. Volz, *Microscale and nanoscale heat transfer*, Topics in Applied Physics, 107, 2007.
[3] S. Volz and G. Chen, *Molecular-dynamics simulation of thermal conductivity of silicon crystals*, Phys. Rev. B, 61:2651:2656, 2000.
[4] G. Domingues, S. Volz, K. Joulain and J.J. Greffet, *Heat transfer between two nanoparticles through near field interaction*, Phys. Rev. Lett., 94:85901, 2005.

[5] P.L. Freddolino, A.S. Arkhipov, S.B. Larson, A. McPherson, and K. Schulten, *Molecular dynamics simulation of the complete satellite tobacco mosaic virus*, Structure, Volume 14, Issue 3, pp.437-449, 2006.
[6] S.Y. Liem, D. Brown, and J.H.R. Clarke, *Molecular dynamics on distributed memory machines*, Computer Physics Communications, Vol. 67, Issue 2, pp. 261-267, 1991.
[7] D. Brown, J.H.R. Clarke, M. Okuda, and T. Yamazaki, *A domain decomposition parallelization strategy for molecular dynamics simulations on distributed memory machines*, Computer Physics Communications, Vol. 74, Issue 1, pp. 67-80, 1993.
[8] J. Ilnytskyi and M.R. Wilson, *A domain decomposition molecular dynamics program for the simulation of flexible molecules with an arbitrary topology of Lennard-Jones and/or Gay-Berne sites*, Computer Physics Communications, Vol. 134, pp. 23-32, 2001.
[9] M.R. Wilson, M.P. Allen, M.A. Warren, A. Sauron, and W. Smith, *Replicated data and decomposition domain molecular dynamics techniques for simulation of anisotropic potentials*, Journal of Computer Chemistry, Vol. 18, pp. 478-488, 1997.
[10] W. Smith, *Molecular dynamics on hypercube parallel computers*, Computer Physics Communications, Vol. 62, pp. 229-248, 1991.
[11] T.C. Germann et al, *25 Tflop/s multibillion-atom molecular dynamics simulations and visualization/analysis on BlueGene/L*, to appear in Supercomputing 2005.
[12] J. Nievergelt, *Parallel methods for integrating ordinary differential equations*, Comm. ACM, Vol. 7, pp. 731-733, 1964.
[13] W.L. Miranker and W. Liniger, *Parallel methods for the numerical integration of ordinary differential equations*, Math. Comp., Vol. 91, pp. 303-320, 1967.
[14] E. Lelarasmee, A.E. Ruehli, and A.L. Sangiovanni-Vincentelli, *The wave-form relaxation method for time-domain analysis of large scale integrated circuits*, IEEE Trans. on CAD of IC and Syst., Vol. 1, pp. 131-145, 1982.
[15] J.-L. Lions, Y. Maday, and G. Turinici, *Résolution d'EDP par un schéma en temps pararéel*, C. R. Acad. Sci. Paris, Série I 332 7 (2001), pp. 661–668.
[16] G. Bal, *On the convergence and the stability of the parareal algorithm to solve partial differential equations*, Lect. Notes Comput.Sci. Eng., Vol. 40, pp. 425–432, Springer, Berlin, 2005.
[17] M.J. Gander and S. Vanderwalle, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput. 29(2), pp. 556–578, 2007.
[18] M.J. Gander and E. Hairer, *Nonlinear convergence analysis for the parareal algorithm*, Domain Decomposition Methods in Science and Engineering XVII, Lecture Notes in Computational Science and Engineering 60, Springer-Verlag, 45-56 (2007).
[19] L. Baffico, S. Bernard, Y. Maday, G. Turinici, and G. Zérah, *Parallel-in-time molecular-dynamics simulations*, Phys. Rev. E 66, 057701 (2002).
[20] B.C. Daly, H. Maris, K. Imamura, and S. Tamura, *Molecular dynamics calculation of the thermal conductivity of superlattices*, Phys. Rev. B 66, 024301 (2002).
[21] E. Hairer, C. Lubich, and G. Wanner, *Geometric Numerical Integration, Structure-Preserving Algorithms for Ordinary Differential Equations*, Springer Series in Computational Mathematics, 2002.
[22] E. Hairer, *Equations Différentielles Ordinaires*, Chapter III, Numerical Analysis lesson, 2004-2005.